**ASSIGNMENT-4**

**DISTANCE DETECTION USING ULTRASONIC SENSOR**

**QUESTION:** Write code and connections in wokwi for ultrasonic sensor. Whenever distance is less than 100 cms send "alert" to ibm cloud and display in device recent events.

**CODE:**

#include <WiFi.h>

#include <PubSubClient.h>

#include <ArduinoJson.h>

WiFiClient wifiClient;

#define ORG "wrcdxi"

#define DEVICE\_TYPE "smps"

#define DEVICE\_ID "1234"

#define TOKEN "smps@123"

#define speed 0.034

char server[] = ORG ".messaging.internetofthings.ibmcloud.com";

char publishTopic[] = "iot-2/evt/Data/fmt/json";

char topic[] = "iot-2/cmd/home/fmt/String";

char authMethod[] = "use-token-auth";

char token[] = TOKEN;

char clientId[] = "d:" ORG ":" DEVICE\_TYPE ":" DEVICE\_ID;

PubSubClient client(server, 1883, wifiClient);

void publishData();

const int trigpin=13;

const int echopin=12;

String command;

String data="";

long duration;

int dist;

void setup()

{

Serial.begin(115200);

pinMode(trigpin, OUTPUT);

pinMode(echopin, INPUT);

wifiConnect();

mqttConnect();

}

void loop() {

publishData();

delay(500);

if (!client.loop()) {

mqttConnect();

}

}

void wifiConnect() {

Serial.print("Connecting to "); Serial.print("Wifi");

WiFi.begin("Wokwi-GUEST", "", 6);

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

Serial.print("WiFi connected, IP address: ");

Serial.println(WiFi.localIP());

}

void mqttConnect() {

if (!client.connected()) {

Serial.print("Reconnecting MQTT client to "); Serial.println(server);

while (!client.connect(clientId, authMethod, token)) {

Serial.print(".");

delay(1000);

}

initManagedDevice();

Serial.println();

}

}

void initManagedDevice() {

if (client.subscribe(topic)) {

Serial.println(client.subscribe(topic));

Serial.println("subscribe to cmd OK");

} else {

Serial.println("subscribe to cmd FAILED");

}

}

void publishData()

{

digitalWrite(trigpin,LOW);

digitalWrite(trigpin,HIGH);

delayMicroseconds(10);

digitalWrite(trigpin,LOW);

duration=pulseIn(echopin,HIGH);

dist=duration\*speed/2;

if(dist<100){

DynamicJsonDocument doc(1024);

String payload;

doc["Distance Alert:"]=dist;

serializeJson(doc, payload);

delay(3000);

Serial.print("\n");

Serial.print("Sending payload: ");

Serial.println(payload);

if (client.publish(publishTopic, (char\*) payload.c\_str())) {

Serial.println("Publish OK");

} else {

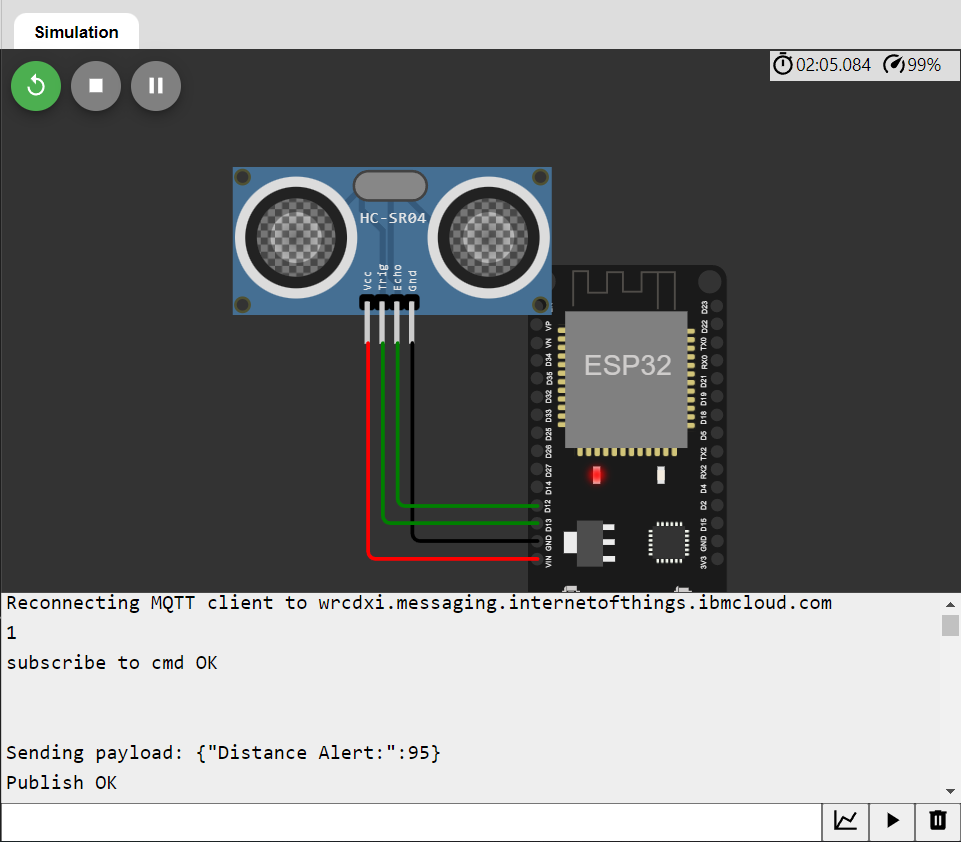
Serial.println("Publish FAILED");

}

}

}

**OUTPUT:**

****